WEST

Generate Collection

L7: Entry 2 of 5 File: USPT May 8, 2001

DOCUMENT-IDENTIFIER: US 6230164 B1
TITLE: Communication system with rapid database synchronization

DEPR:
FIG. 1a illustrates a block diagram of a telephone system 10 with PHS capabilities. The PSTN (public switched telephone network) 12 is coupled to a plurality of devices 14, such as telephones, PBXs (private branch exchanges), modems, and digital devices. Additionally, a plurality of PHS base stations 16 are connected to the PSTN 12. PHS handsets (or other devices such as digital modems) 18 communicate with other devices coupled to the PSTN 12 via the base stations 16 using wireless communication.

DEPR:
AIN (Advanced Intelligent Network) system 22 includes one or more STPs (signal transfer points) 24 coupled to the PSTN 12. The STPs 24 are coupled to one another and to a plurality of SCP (service control point) pairs 26. Each SCP pair 26 comprises two fully redundant SCPs 26a and 26b, which are described in greater detail herein below. STPs 24 are also connected to an NCC (network control center) 28, an SMS (service management system) 30 and VMS (voice mail system) 32. NCC 28, SMS 30 and VMS 32 are coupled to SCP pairs 26. NCC 28 includes a CGTT (Centralized Global Title Table) 34.

DEPR:
SCPs 26 may be coupled to SMS 30 via dedicated point-to-point X.25 links. SCPs 26a and 26b of a pair are generally physically located in different cities and may be coupled together via some communications line such as a point-to-point wide area network (WAN) link or a Media Access Control (MAC) bridge.

DEPR:
Some exemplary messages that are transmitted between SMS 30, SCPs 26, and base station 16 are shown in FIG. 1b. When a new subscriber using a portable handset 18 is being added to communications network 10, SMS 30 issues an INSERT command to add a new unique personal subscriber number or telephone number to both SCPs 26a and 26b in the appropriate pair as defined by the CGTT 34. A subscriber who no longer desires wireless service can be deleted in a similar manner with DELETE messages to both SCPs 26a and 26b. SMS 30 may also issue UPDATE messages to provide information, such as add a new service, to SCPs 26a and 26b. These messages are examples of static data updates.

DEPR:
FIG. 2 provides a more detailed block diagram of a SCP 26b coupled to its mate SCP 26a constructed according to the teachings of the present invention. Each SCP includes an active Platform Manager (PM) 34 and a standby Platform Manager 36 coupled by a bus, local area network (LAN), or a local area network hub 50 to a predetermined number of Application Processor Groups (APG.sub.1 -APG.sub.m) 38-42. To provide greater network integrity and fault tolerance, dual LAN or hubs may be used to connect the PMs to the APGs to provide redundancy. Each APG 38-42 includes a plurality of Intelligent Processor Units (IPU.sub.1 -IPU.sub.n) 4448. One or more IPU may be configured as a spare or standby IPU that may be brought on-line as another IPU fails. A host 51 interfaces between the STPs 24 and the IPUs of the SCP. A route table, described below, directs queries to the correct IPU. The route table is managed by the PMs and is distributed to the host 51 and to the IPUs. By

distributing the route table to the host 51 and the IPUs, queries from the STPs can be quickly routed to the correct IPU.

DEPR:
Referring to FIG. 3, it may be seen that each Platform Manager 34 and 36 includes a PM Database Manager process 52 and an APG Database Manager process 54 for each APG. Each IPU.sub.1, 60-64 also has an IPU Database Manager process 66-70 and shared memory 72-76 residing therein. Shared memory 72-76 may be implemented by any fast memory device, including random access memory (RAM) devices, and is accessible to all the processes residing in the IPUs. A pair of mirrored memory storage devices 80 and 82 are coupled to each IPU 60-64, where IPUs 60-64 may all access the memory storage devices 80 and 82 simultaneously. Simultaneous file access may be accomplished by implementing memory storage devices 80 and 82 with multi-port media, or by running IPUs 60-64 in a multi-initiator mode to each memory device 80 and 82. Memory storage devices 80 and 82 may be implemented with solid state disks or any other suitable storage media. In the multi-initiator mode, memory storage devices 80 and 82 may each be coupled to IPUs 60-64 by a separate bus or Small Computer Systems Interface (SCSI). Constructed and configured in this manner, any one of IPUs 60-64 has access to both memory storage devices 80 and 82.

DEPR:
Memory storage devices 80 and 82 may be segmented into a predetermined partitions or filesystems, where X of them are used to store subscriber files. The portable handset subscriber database is comprised of a fixed number of files which are stored on mirrored disks of APGs 3842 of SCP 30, where there is a pair of mirrored disks per APG. A subset of subscriber records in the entire subscriber database is assigned to each subscriber file. Each subscriber file is assigned to be stored in a specific filesystem on a specific pair of mirrored disks in the SCP, such that each APG services an exclusive subset of the subscriber database. As shown in FIG. 3, the number of files that may be stored on a pair of disks is Y. The pair of disks are mirrored, so that the contents of the disks, if both are operational, are always the same.

DEPR:
A third table, route table 94, is also maintained by PM Database Manager process 52. Route table 94 contains information for each file in the database. It is used to supply the routing information to the host (see FIG. 2), such as a Message Transport Network (MTN), coupled to the PMs, so that the host may direct queries to the appropriate IPU depending on each IPU's database load. Route table may include:

DEPR:
Referring to FIG. 6, IPU Database Manager 66 may include a number of objects, such as an IPU Database Handler 130 which provides an interface to APG Database Manager and the application processes on IPU node 60-64 (FIG. 3). IPU Database Manager 66 is also responsible indirectly for mounting and unmounting filesystems on the IPU node and mapping and unmapping the database files to and from shared memory 72 (FIG. 3). Process 66 Object 130 also communicates new database load information to the application processes on the node.

DEPR:
A Group File Handler 132 is an object that is responsible for periodically synchronizing the database files that are in shared memory 72 (FIG. 3) to the mirrored disks 80 and 82 (FIG. 3). An IPU Disk Manager object 134 is instantiated by IPU Database Handler 130 and is responsible for performing the mounting and unmounting of the filesystems. A Database File Mapper object 136 is responsible for mapping and unmapping files to and from shared memory. There is one Database File Mapper 136 per file on the IPU node. A Subscriber Database Access object 138 is responsible to provide processes on remote nodes access to the portion of the database handled by this particular IPU. Remote nodes include nodes residing on mate SCP 26a (FIG. 2), for example.

DEPR:

APG Database Manager 52 first instantiates an APG Database Manager 54 for each APG in the SCP. FIG. 7 is an exemplary process flow for APG Database Manager initialization beginning in block 160. First, an APG Database Handler object 110 is instantiated, as shown in block 162. In block 164, APG Database Handler 110 then instantiates Database Route Control 112, Database Config Table Access 116, and IPU Info Table 114. Database Route Control object 112 then instantiates and initializes all the tables 90-94 in APG Database Manager 52, as shown in blocks 166 and 168. If the PM is active, as determined in block 170, then an audit is performed of IN_SERVICE IPU(s) by APG Database Handler 96 in block 172. The audit yields the database load(s) of audited IPU(s), which is used to update the tables with this information, as shown in block 174. Subsequently in blocks 176 and 178 APG Database Manager 54 registers itself with the PM node process before ending the initialization process. The act of registration reveals the object's instance to other processes, so that others may communicate therewith.

DEPR:
FIG. 8 illustrates an exemplary process flow for IPU Database Manager initialization 190. In block 192, instances of IPU Database Handler 130, Group File Handler 132 and Subscriber Database Access 138 objects are instantiated. A sync timer used for shared memory-to-disk updating is initiated in block 194. IPU Database Handler 130 then requests for its share of the database load from APG Database Handler 110, as shown in block 196. In response, APG Database Manager 54 looks up database configuration and IPU tables for information on the filesystems and the requesting IPU, with this information, IN_SERVICE IPU database loads are determined based on the number of IPUs that are IN_SERVICE and traffic conditions, as shown in blocks 198 and 200. Database loads are distributed to the requesting IPU in block 202. IPU Database Manager 66 then registers itself with the PM node process, as shown in block 206. IPU Database Manager then receives the load, as shown in block 204. The filesystem(s) belonging to the portion of the database that are assigned to the IPU are then added or mounted to the IPU, as shown in block 208. The initialization process subsequently ends in block 210.

DEPR:
FIG. 9 shows the process flow in the APG Database Manager when a Platform Manager 34 transitions from the standby mode to the active mode, beginning in block 230. All the APG Database Managers 54 operating on the Platform Manager perform an audit of their IPU database loads, as shown in block 232. Database Route Controls 112 of each APG then initializes all tables, including database config table 90, route table 94, and IPU table 92. APG Database Handler 110 then obtains a list of IN_SERVICE IPU(S) for its APG, and queries each IN_SERVICE IPU for its database load, as shown in blocks 236 and 238. The tables are reconstructed and updated with the information supplied by the IN_SERVICE IPUs, as shown in block 240. Also dependent on this audit information, unassigned filesystem(s) are assigned to those IN_SERVICE IPU(s) that are lightly loaded, and IPU(s) with no load assigned are assigned their default database load, as shown in blocks 242 and 244. The new database load distribution results in new routing information in route table 94, which is provided to the host by APG Database Handlers 110. The standby-to-active transition process ends in block 248.

DEPR:
IPU failures are handled by the process flow shown in FIG. 10 beginning in block 250. In block 252, APG Database Manager 54 receives notification of an IPU failure from the PM node process. A timer is set for each failed IPU, as shown in block 254. If APG Database Manager 54 receives an IPU IN_SERVICE notification prior to the timer's expiration, as determined in block 256, then nothing more needs to be done. However, if no such notification was received, and if an IPU exit notification is received or if the timer expires, as shown in block 258, the load carried by the failed IPU is reassigned and sent to the remaining IN_SERVICE IPUs, as shown in blocks 260 and 262. If any additional IN_SERVICE IPUs fail now, as determined in block 264, execution proceeds to block 260, where the database loads are again redistributed to the remaining IN_SERVICE IPUs. If no additional IPUs fail, as determined in block 264, then

Database Route Control 112 extracts updated routing information from route table 94 and APG Database Handler provides this information to the host, as shown in blocks 266 and 268. The process ends in block 270.

DEPR:
To add filesystem(s) to an IPU, the exemplary process flow beginning in block 280 and shown in FIG. 11 may be used. IPU Disk Manager 134 mounts the filesystem(s) to be added to the appropriate IPU, as shown in block 282. The files in the mounted filesystem(s) are then mapped to shared memory by Group File Handler 132, as shown in block 284. Subscriber Database Access 138 then attaches to the shared memory files, as shown in block 286. Because records in the files are organized and searchable by accessing pointers in a Red-Black Tree data structure in the preferred embodiment, the Red-Black tree is corrected or rebuilt, if necessary. A Red-Black Tree is a balanced tree data structure that facilitates quick searches, where all the records in a file may be located by searching the nodes in the Red-Black Tree. Recall that the modulo operation yields the file index, and by searching the appropriate Red-Black Tree shared memory file, the specific record may be accessed. It is important to acknowledge that other data structures may be used without departing from the spirit of the invention. Thereafter, Subscriber Database Access 138 sends messages to all concerned applications about the new IPU file load, as shown in block 290. The process then ends in block 292.

DEPR:
FIG. 15 shows the process flow for beginning database reconfiguration, beginning in block 360. If database reconfiguration is desired, the appropriate Reconfig Flag is set for the APG, as shown in block 362. Next, a retry counter or timer (RETRY_CNT) is reset to zero, as shown in block 364. Execution then enters a loop in which the reconfiguration process waits for load balancing to complete if it is in progress. The retry counter is first checked to see if it has reached a predetermined upper limit, for example 180, as shown in block 368. If the upper limit has been reached, it is determined that the PM node has failed and its status is downgraded to the OS_MIN state. If the retry count has not yet reached the predetermined upper limit, then the Load Balance Flag is checked to see if it is set, as shown in block 370. If it is not set, then execution may proceed with database reconfiguration. Otherwise, the retry counter is incremented and a predetermined amount of time, for example one second, is permitted to elapse before returning to the beginning of the loop at block 366.

DEPR:
There are several data synchronization processes taking place in distributed redundant database 10. The data stored in the shared memory of each IPU is synchronized to both mirrored disks, and all modified transient data in the database of each SCP is provided to its mate SCP.

DEPR:
FIG. 16 is an exemplary process flow 380 for synchronizing the data in the IPUs' shared memory 72-76 (FIG. 3) to mirrored disks 80 and 82 (FIG. 3). In block 382, the IPU sync timer is checked to determined whether it has expired. Recall that this timer was initialized during IPU Database Manager initialization, as shown in block 194 in FIG. 8. If the sync timer has not yet expired, a predetermined amount of time is permitted to elapse, and the timer is rechecked, until the sync timer is expired. The expiration of the sync timer indicates that it is time to copy a portion or block of a file in the shared memory to the mirrored disks, as shown in block 384. The sync timer is then reset, as shown in block 386, and execution returns to block 382. At the expiration of the sync timer next time, the next portion of the file is copied to disk. When an entire file has been copied, the next file is copied to disk. In this manner, all the files in the shared memory of each IPU are copied to disk. Because each IPU is assigned a different set of filesystems, the IPUs may "sync" to disk in parallel in the multi-initiated mode without interfering with each other's operations. It may be noted that this "sync" to disk process primarily updates the disks with transient data, such as subscriber current location. Static data such as adding or deleting new subscribers, service

option updates, and subscriber preference data are immediately written to the mirrored disks generally simultaneously with writing the same data to the shared memory.

DEPR:
Within the fourth and fifth fields, there are nine sub-fields. The first sub-field defines the translation type. If necessary, this can be used to identify different network types. The second sub-field identifies a number plan for numbering plan, which may vary between providers. The third sub-field defines a backup mode, either to the first SCP, load sharing between the first and second SCPs, or to the second SCP if the first SCP is inoperable. The fourth, fifth and sixth sub-fields identify whether or not the STP is the final STP, the name of the primary SCP and the destination application in the primary SCP. The seventh, eighth and ninth sub-fields identify the same information for the backup path.